

Claim 1. A method for use by a host microprocessor which translates sequences of instructions from a target instruction set for a target processor to sequences of instructions for the host microprocessor comprising the steps of:

beginning execution of a first sequence of target instructions by committing state of the target processor and storing memory stores generated by previously-executed sequences of instructions at a point in the execution of instructions at which state of the target processor is known,

beginning execution of a speculative sequence of host instructions following a branch from the first sequence of target instructions by immediately committing state and storing memory stores,

attempting to execute the speculative sequence of host instructions until another point in the execution of target instructions at which state of the target processor is known,

rolling back to last committed state of the target processor and discarding memory stores generated by the speculative sequence of host instructions if execution fails, and

beginning execution of a next sequence of target instructions if execution succeeds.

Claim 2 A method as claimed in Claim 1 including an additional step of releasing a lock for any sequence of host instructions running in a locked condition immediately after committing state of the target

A8

processor and storing memory stores generated by previously-executed translation sequences.

Claim 3. A method for use by a host microprocessor which translates sequences of instructions from a target instruction set for a target processor to sequences of instructions for the host microprocessor comprising the steps of:

beginning execution of a first sequence of target instructions by committing state of the target processor and storing memory stores generated by previously-executed sequences of instructions at a point in the execution of instructions at which state of the target processor is known,

executing a sequence of host instructions from the first sequence of target instructions commencing immediately after committing state of the target processor and storing memory stores previously generated by a execution until another point in the translation of target instructions at which state of the target processor is known,

beginning execution of a next sequence of target instructions by committing state of the target processor and storing memory stores generated by the execution of the first sequence of target instructions at a point in the execution of target instructions at which that state is known, and

executing a sequence of host instructions from the next sequence of target instructions commencing immediately after committing state of the target processor and storing memory stores generated by the execution of

23 the first sequence of target instructions until another point in the
24 execution of target instructions at which state of the target processor is
25 known.

1 Claim 4. A method for use by a host microprocessor which translates
2 sequences of instructions from a target instruction set for a target
3 processor to sequences of instructions for the host microprocessor
4 comprising the steps of:

5 translating a first speculative sequence of host instructions from the first
6 sequence of target instructions from a point in the translation of target
7 instructions at which state of the target processor is known,

8 ending the first sequence of target instructions in response to
9 encountering a branch from the first sequence in the target program by:

10 branching to a branch sequence of target instructions,

11 committing state of the target processor and storing memory stores
12 generated by the first translation sequence after a branch taken

13 before executing a branch sequence of host instructions, and

14 ending execution of the first sequence of target instructions if a branch is
15 not taken from the first sequence by:

16 rolling back to last committed state of the target processor and

17 discarding memory stores generated by the speculative sequence of
18 host instructions if execution fails, and

committing state of the target processor and storing memory stores
generated by the first sequence at the end of the sequence of target
instructions at which state of the target processor is known.

Figure 1. Schematic representation of the experimental design. The subjects were divided into two groups: the control group (CG) and the experimental group (EG). The CG was divided into two subgroups: the control group (CG) and the control group (CG). The EG was divided into two subgroups: the experimental group (EG) and the experimental group (EG). The subjects were divided into two groups: the control group (CG) and the experimental group (EG). The CG was divided into two subgroups: the control group (CG) and the control group (CG). The EG was divided into two subgroups: the experimental group (EG) and the experimental group (EG).